PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-143764
(43)Date of publication of application : 28.05.1999

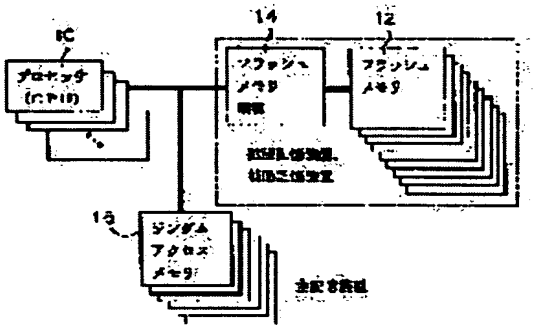
(51)Int.Cl. G06F 12/02

(21)Application number : 09-317612
(22)Date of filing : 04.11.1997

(71)Applicant : VICTOR CO OF JAPAN LTD
(72)Inventor : TANAKA KAZUYA

(54) FLASH TYPE MEMORY, ITS MANAGEMENT DEVICE, STORAGE DEVICE AND COMPUTER SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To execute program on an extension storage device without loading an execution program to a main storage device by providing a block of a flash type memory with a header area which stores management information of the block and a storage area which stores a data file that is divided into a block unit.
SOLUTION: This flash type memory is applied to a computer system which includes a processor 10, an extension or auxiliary storage device which includes flash memory 12 and its controller 14 and a RAM (main storage device) 16. A management unit of the memory 12 is one block that is a minimum erase (erasure) unit in order to manage an execution program area and a data area. The storage area of each block stores the execution program area or the data area, a header area which has information that manages the block and a data file which is divided into a block unit.



LEGAL STATUS

[Date of request for examination] 28.09.2000
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number] 3503448

[Date of registration]	19.12.2003
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－143764

(43)公開日　平成11年(1999)５月28日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　12/02	５１０	Ｇ０６Ｆ　12/02	５１０Ａ

審査請求　未請求　請求項の数15　ＦＤ　（全 18 頁）

(21)出願番号　特願平9－317612

(22)出願日　平成９年(1997)11月４日

(71)出願人　000004329
日本ピクター株式会社
神奈川県横浜市神奈川区守屋町３丁目12番地

(72)発明者　田中　和也
神奈川県横浜市神奈川区守屋町３丁目12番地　日本ピクター株式会社内

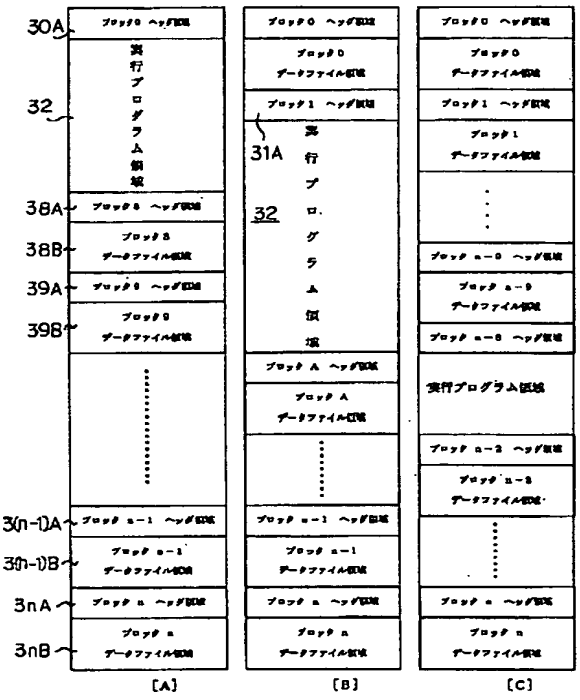
(74)代理人　弁理士　梶原　康稔

(54)【発明の名称】　フラッシュ型メモリ，その管理装置，記憶装置，コンピュータシステム

(57)【要約】

【課題】　ブロック消去方式のフラッシュ型メモリを使用してプログラムの実行を可能とするとともに、効率的なメモリ管理を行う。

【解決手段】　実行プログラム領域は、連続する複数のブロックに確保される。複数の連続する物理ブロックに確保された実行プログラム領域は、１つのヘッダ領域によって管理される。データファイル領域は、例え複数の連続する物理ブロックに確保されたとしても、各ブロック毎にヘッダ領域によって管理される。［Ａ］では、連続する８個の物理ブロック０～７を実行プログラム領域として割り当て、それ以外の物理ブロック８～ｎをデータファイル領域として割り当てている。実行プログラム領域３２を管理する物理ブロックヘッダ３２Ａは、物理ブロック０にのみ存在する。物理ブロック８～ｎには、それぞれヘッダ領域３８Ａ～３ｎＡが存在し、ブロック毎に管理される。



【特許請求の範囲】

【請求項１】　ブロック単位で記憶内容を消去できるフラッシュ型メモリのブロックに、そのブロックの管理情報を記憶するヘッダ領域と、ブロック単位に分割されたデータファイルを格納する記憶領域とを備えたことを特徴とするフラッシュ型メモリ。

【請求項２】　ブロック単位で記憶内容を消去できるフラッシュ型メモリが第１，第２，及び第３のブロックを含んでおり、

第１のブロックは、そのブロックの管理情報を記憶するヘッダ領域と、ブロック単位に分割されたデータファイルを格納する記憶領域とを備えており、

第２のブロックは、ブロック単位に分割された実行プログラムファイルを格納する記憶領域を備えており、

第３のブロックは、実行プログラムファイルが格納されたブロックの管理情報を記憶するヘッダ領域と、ブロック単位に分割された実行プログラムファイルを格納する記憶領域とを備えていることを特徴とするフラッシュ型メモリ。

【請求項３】　前記ブロックの管理情報として、データファイル又は実行プログラムファイルのいずれがそのブロックに格納されているかを示す情報が含まれることを特徴とする請求項１又は２記載のフラッシュ型メモリ。

【請求項４】　前記ブロックの管理情報として、そのブロックに対する消去回数が含まれることを特徴とする請求項１，２，又は３のいずれかに記載のフラッシュ型メモリ。

【請求項５】　前記ブロックの管理情報として、ブロックを消去した順番を示すイレースシーケンスナンバが含まれることを特徴とする請求項４記載のフラッシュ型メモリ。

【請求項６】　前記ブロックの管理情報として、ブロックを消去した時刻情報が含まれることを特徴とする請求項４又は５記載のフラッシュ型メモリ。

【請求項７】　前記ブロックの管理情報として、そのブロックが有効／遷移中／無効／初期化のいずれの状態にあるかを示す情報が含まれることを特徴とする請求項１，２，３，４，５，又は６のいずれかに記載のフラッシュ型メモリ。

【請求項８】　ブロック単位に分割されたデータファイルを各ブロックに割り当てるとともに、各ブロックの管理情報を該当するブロックにそれぞれ格納する手段を備えたことを特徴とする請求項１記載のフラッシュ型メモリの管理装置。

【請求項９】　隣接する複数の前記第２及び第３のブロックに対して実行プログラムファイルの割り当てるとともに、それら実行プログラムファイルが割り当てられたブロックの管理情報を前記第３のブロックに格納する手段を備えたことを特徴とする請求項２記載のフラッシュ型メモリの管理装置。

【請求項１０】　ブロックをイレースするための候補ブロックを選択する際に、消去回数が同じであれば、前記イレースシーケンスナンバに基づいて消去した順番に優先的にブロックを選択する手段を備えたことを特徴とする請求項５記載のフラッシュ型メモリ管理装置。

【請求項１１】　ブロックをイレースするための候補ブロックを選択する際に、消去回数が同じであれば、前記時刻情報に基づいて消去したブロックの時刻が古いブロックを優先的に選択する手段を備えたことを特徴とする請求項６記載のフラッシュ型メモリ管理装置。

【請求項１２】　各ブロックに格納されているデータの書換回数を低減する手段を備えたことを特徴とする請求項８，９，１０，又は１１のいずれかに記載のフラッシュ型メモリ管理装置。

【請求項１３】　システムの復旧やデータの復元を行なう手段を備えたことを特徴とする請求項８，９，１０，１１，又は１２のいずれかに記載のフラッシュ型メモリ管理装置。

【請求項１４】　請求項１，２，３，４，５，６，又は７のいずれかに記載のフラッシュ型メモリと、請求項８，９，１０，１１，１２，又は１３のいずれかに記載の管理装置とを備えたことを特徴とする記憶装置。

【請求項１５】　請求項１４記載の記憶装置を備えており、前記実行プログラムファイルをフラッシュメモリ上で動作させることを特徴とするコンピュータシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は、フラッシュ型メモリ，その管理装置，記憶装置，コンピュータシステムにかかり、更に具体的には、フラッシュ型メモリの効率的な管理手法の改良に関するものである。

【０００２】

【背景技術】一般にフラッシュタイプのフローティングゲートトランジスタを含む電気的に消去可能なプログラマブル読出専用メモリ（ＥＥＰＲＯＭ）は、現在市場で容易に入手できる。これらのいわゆるフラッシュメモリは、機能・性能面でＥＰＲＯＭメモリと類似した不揮発メモリであり、メモリ内に分割されているブロックを消去する回路内プログラマブル動作を可能にするという機能を更に有する。フラッシュメモリでは、以前に書き込まれたメモリの領域を前もって消去することで、その書き換えが行われる。

【０００３】典型的なコンピュータシステムでは、オペレーティングシステム（以下、単に「ＯＳ」という）のプログラムがそのシステムのデータ記憶装置のデータ管理を担う。ＯＳプログラムとの互換性を達成するために必要かつ十分であるデータ記憶装置のアトリビュート（属性）は、データ記憶装置のいかなる位置からもデータを読み出すことができ、これにデータを書き込むことができることである。しかし、フラッシュメモリの場

合、データが既に書き込まれている領域には、その領域のデータを消去しなければデータを書き込むことはできない。このため、フラッシュメモリは、典型的な既存のＯＳプログラムとは互換性がない。

【０００４】このような点に着目し、既存のＯＳプログラムによってフラッシュメモリを管理することを可能にするソフトウエアが先行技術において提案されている。この先行技術のプログラムでは、フラッシュメモリを「書込み１回読出し複数回」の装置として動作させるか、「書込み複数回読出し複数回」の装置として動作させている。前者は、以前に書き込まれているメモリ場所を再利用することはできない装置であり、補助記憶装置や拡張記憶装置として使用できる。後者は、以前に書き込まれているメモリ領域を再利用可能とし、その領域中にはフラッシュメモリの書換回数を少なくするような制御を持つ補助記憶装置（半導体ファイル記憶装置）がある。

【０００５】

【発明が解決しようとする課題】しかしながら、以上のような背景技術には、次のような不都合がある。

(1)上述のような先行技術に見られる装置によれば、フラッシュメモリを使用した拡張記憶装置上では書換回数を少なくなるような制御構造を持たずに実行プログラムを動作させている。つまり、フラッシュメモリに対して新たにデータの書換えを行うときは、全ブロックを一括して消去し、その後、データを記憶させる必要がある。また、マルチタスク，マルチスレッド，マルチプロセスに代表される並列実行プログラムやデータの共有あるいは占有という管理を行う場合において、一般のMMU（メモリ管理ユニット）を駆使してフラッシュメモリをブロック（ページ）毎に管理する方式では、書換回数の管理と書換回数の低減を図ることは困難である。

【０００６】(2)フラッシュメモリを使用した拡張記憶装置上でプログラムの実行を可能とする装置にはないものの、フラッシュメモリの書換回数を少なくするような制御構造を持つ装置として補助記憶装置（半導体ファイル記憶装置）がある。補助記憶装置において書換回数を少なくするためにその情報（書換回数テーブル）を異なるメモリ上に記憶する方式である。しかし、実行プログラムを主記憶装置にロードすることなく、フラッシュメモリ上で動作させるプログラムにおいて、単純に同一フラッシュメモリ上に記憶する方式では、ブロック間にまたがるデータを無駄なく完全に連続的な配置を可能にすることは困難である。

【０００７】この発明は、以上の点に着目したもので、その目的は、フラッシュメモリを使用した拡張記憶装置上でフラッシュメモリの書換回数を少なくするような制御構造を持つ場合に、主記憶装置に実行プログラムをロードすることなく拡張記憶装置上でプログラムの実行を可能とすることである。他の目的は、同一のフラッシュメモリを使用し、補助記憶装置として書換回数を少なくする制御構造を持ちながら、マルチプロセッシング，マルチスレッド，マルチタスクといった高度なプログラムの実行環境においても、メモリ管理装置によって、安全に実行プログラムのモジュールやデータファイルの管理の実現を可能とすることである。更に他の目的は、同一フラッシュメモリ上で、拡張記憶装置および補助記憶装置の混在を可能とするファイル管理技術を提供することである。

【０００８】

【課題を解決するための手段】本発明のフラッシュ型メモリは、ブロック単位で記憶内容を消去できるフラッシュ型メモリのブロックに、そのブロックの管理情報を記憶するヘッダ領域と、ブロック単位に分割されたデータファイルを格納する記憶領域とを備えたことを特徴とする。あるいは、ブロック単位で記憶内容を消去できるフラッシュ型メモリが第１，第２，及び第３のブロックを含んでおり、第１のブロックは、そのブロックの管理情報を記憶するヘッダ領域と、ブロック単位に分割されたデータファイルを格納する記憶領域とを備えており、第２のブロックは、ブロック単位に分割された実行プログラムファイルを格納する記憶領域を備えており、第３のブロックは、実行プログラムファイルが格納されたブロックの管理情報を記憶するヘッダ領域と、ブロック単位に分割された実行プログラムファイルを格納する記憶領域とを備えていることを特徴とする。

【０００９】主要な形態の一つによれば、前記ブロックの管理情報として、(1)データファイル又は実行プログラムファイルのいずれがそのブロックに格納されているかを示す情報，(2)そのブロックに対する消去回数，(3)ブロックを消去した順番を示すイレースシーケンスナンバ，(4)ブロックを消去した時刻情報，(5)そのブロックが有効／遷移中／無効／初期化のいずれの状態にあるかを示す情報，の少なくとも一つが含まれる。

【００１０】本発明のメモリ管理装置は、ブロック単位に分割されたデータファイルを各ブロックに割り当てるとともに、各ブロックの管理情報を該当するブロックにそれぞれ格納する手段を備えたことを特徴とする。あるいは、隣接する複数の前記第２及び第３のブロックに対して実行プログラムファイルの割り当てるとともに、それら実行プログラムファイルが割り当てられたブロックの管理情報を前記第３のブロックに格納する手段を備えたことを特徴とする。

【００１１】主要な形態の一つによれば、ブロックをイレースするための候補ブロックを選択する際に、(1)消去回数が同じであれば、前記イレースシーケンスナンバに基づいて消去した順番に優先的にブロックを選択する手段，(2)消去回数が同じであれば、前記時刻情報に基づいて消去したブロックの時刻が古いブロックを優先的に選択する手段，の少なくとも一つを備えたことを特徴

とする。

【００１２】他の形態によれば、(1)各ブロックに格納されているデータの書換回数を低減する手段，(2)システムの復旧やデータの復元を行なう手段，の少なくとも一つを備えたことを特徴とする。

【００１３】本発明の記憶装置は、前記いずれかのフラッシュ型メモリと、前記いずれかの管理装置とを備えたことを特徴とする。本発明のコンピュータシステムは、前記記憶装置を備えており、前記実行プログラムファイルをフラッシュメモリ上で動作させることを特徴とする。

【００１４】この発明の前記及び他の目的，特徴，利点は、以下の詳細な説明及び添付図面から明瞭になろう。

【００１５】

【発明の実施の形態】以下、本発明の実施の形態について詳細に説明する。本形態は、例えば図１に示すように、プロセッサ１０，フラッシュメモリ１２およびその制御装置１４を含む拡張もしくは補助の記憶装置，ＲＡＭ（主記憶装置）１６を含むコンピュータシステムに適用される。このようなシステムで、同一フラッシュメモリを使用して、実行プログラムモジュールとデータファイルのデータを共存させるとともに、データの書換回数をできる限り低減するデータの格納方法と管理・制御のための装置を得ようとするものである。

【００１６】まず、実行プログラム領域とデータ領域を管理するため、フラッシュメモリ１２の管理単位をその最小イレース（消去）単位である１ブロックとする。そして、この最小単位であるブロック内に、実行プログラム領域とデータ領域の混在を認めないこととする。各ブロックには、実行プログラム領域もしくはデータ領域と、そのブロックを管理する情報を持つヘッダ領域とが存在する。

【００１７】ここで、データ領域に割り当てようとするブロックには、そのブロックのヘッダ領域内にデータの属性を持つことを宣言する。同様に、実行プログラム領域に割り当てようとするブロックには、そのブロックのヘッダ領域内に実行プログラムの属性を持つことを宣言する。ただし、この実行プログラム領域については、１つのブロックに１つのヘッダ領域が必ずしも必要ということではない。すなわち、連鎖する複数ブロックを連続的に実行プログラム領域として確保したい場合は、確保ブロック数をヘッダ領域内で宣言する。これによって、宣言した連鎖する複数ブロックは、ヘッダ情報を持たなくてもよい。つまり、実行プログラムのメモリ配置として、１つのヘッダ領域により複数の連続するブロックに実行プログラム領域を確保する。一方、データ領域は１つのブロック内にヘッダ領域と混在することになる。このデータ領域内には、該当するブロックに関する情報を付加する。

【００１８】図２には、本形態におけるソフトウエアの基本構造の一例が示されている。同図のように、ＯＳ２０やアプリケーションソフト２２とのソケットインターフェースとして、いわゆるドライバ層２４に相当する部分にフラッシュ管理マネージャ２６を設ける。このフラッシュ管理マネージャ２６は、本形態のシステム用ドライバと考えることができる。このフラッシュ管理マネージャ２６によって、ブロック情報の管理やデータのリード／ライトの制御が行なわれる。

【００１９】例えば、図３［Ａ］に示すように、ブロック消去がｉバイト単位で可能で、かつブロック数がｎ＋１（１６進数）個あるフラッシュメモリが１つあるとする。これら各ブロックを、物理ブロックと呼ぶことにする。そして、各物理ブロックに、アドレスの低い方から高い方へ順に物理ブロックナンバを「０」から「ｎ」まで割り振る。図示の例では、図の上方がアドレスが低く、物理ブロック０，物理ブロック１，物理ブロック２，……，物理ブロックｎとなっている。

【００２０】ここで、それぞれの物理ブロックを管理している情報のヘッダ領域としてｊバイト分を確保したとすると、各物理ブロックに割り当てられる実行プログラム領域あるいはデータファイル領域のサイズは、ｉ－ｊバイトとなる。そして、同一ブロックの記憶領域には、実行プログラム領域とデータファイル領域の混在は認めないことにする。この様子を示すと、図３［Ｂ］のようになる。例えば、最上位の物理ブロック０には、ｊバイトのヘッダ領域３０Ａと、ｉ－ｊバイトの記憶領域３０Ｂが存在する。次の物理ブロック１には、ｊバイトのヘッダ領域３１Ａと、ｉ－ｊバイトの記憶領域３１Ｂが存在する。以下の物理ブロックについても同様である。

【００２１】次に、実行プログラム領域の確保例について、図４を参照しながら説明する。図４［Ａ］には、連続する８個の物理ブロック０～７を実行プログラム領域として割り当て、それ以外の物理ブロック８～ｎをデータファイル領域として割り当てた例である。上述したように、複数の連続する物理ブロックに確保された実行プログラム領域は、１つのヘッダ領域によって管理される。一方、データファイル領域は、例え複数の連続する物理ブロックに確保されたとしても、各ブロック毎にヘッダ領域によって管理される。従って、図４［Ａ］の例では、実行プログラム領域３２の物理ブロックヘッダ３２Ａは物理ブロック０にのみ存在し、物理ブロック１～７には存在しない。そして、後述するように、物理ブロック１～７の管理情報等は、物理ブロック０のヘッダ領域３２Ａにまとめられることになる。一方、物理ブロック８～ｎにはデータファイル領域３８Ｂ～３ｎＢがそれぞれ確保されるが、各ブロックにそれぞれヘッダ領域３８Ａ～３ｎＡが存在し、これらによってブロック毎に管理される。

【００２２】次に、図４［Ｂ］の例は、物理ブロック０をデータファイル領域として割り当て、物理ブロック１

～Ａを実行プログラム領域として割り当て、それ以後はデータファイル領域として割り当てた例である。図４［Ｃ］の例は、物理ブロック０～ｎ－９をデータファイル領域に、物理ブロックｎ－８～ｎ－３を実行プログラム領域に、それ以後をデータファイル領域にそれぞれ割り当てた例である。このように、連続して確保された実行プログラム領域は一つのヘッダ領域によって管理される。そして、物理ブロック数の制限や固定したアドレス配置にすることはない。しかし、データファイル領域は、連続して確保されたとしても、各ブロック毎に管理される。

【００２３】図５に示す例では、実行プログラム領域が分割して配置されている。まず、図５［Ａ］の例では、物理ブロック０～７に実行プログラム領域(1)が割り当てられており、物理ブロック８にはデータファイル領域が割り当てられている。そして、物理ブロック９～Ｆ－１に実行プログラム(2)が割り当てられており、物理ブロックＦ以後はデータファイル領域用に確保されている。この例でも、連続する実行プログラム領域は、一つのヘッダ領域で管理されている。

【００２４】図５［Ｂ］では、物理ブロック０がデータファイル領域用に割り当てられており、物理ブロック１～８が実行プログラム領域(1)に割り当てられている。そして、物理ブロック９～ｎ－９までがデータファイル領域用に割り当てられており、それ以後最後までが実行プログラム領域(2)として割り当てられている。図５［Ｃ］では、物理ブロック０，１がデータファイル領域として割り当てられており、物理ブロック２～４が実行プログラム領域(1)に割り当てられている。そして、その後の物理ブロック５～ｎ－９がデータファイル領域として割り当てられており、物理ブロックｎ－８～ｎ－１が実行プログラム領域(2)として割り当てられており、最後の物理ブロックｎがデータファイル領域に割り当てられている。これら図５では、実行プログラム領域として２つの領域(1)，(2)が指定されている例を示している。しかし、本形態では、実行プログラム領域を更に複数の領域で配置することが可能である。

【００２５】次に、物理ブロックのヘッダ領域について説明する。物理ブロックヘッダには、実行プログラム領域用とデータファイル領域用とがある。この２種類の領域を判断するために、ヘッダは共通部分の構造を持つことが望ましい。そこで、物理ブロックヘッダの共通する部分の構造を、例えば次のように割り当てる。

【００２６】(1)状態（有効／無効／遷移中／初期化完了）を示すフラグ
(2)実行プログラム領域／データファイル領域を示すマーカーフラグ
(3)連続予約ブロックの領域数
(4)ブロックの消去回数
(5)ブロックの消去した順番を示す番号（イレースシーケンスナンバ）
(6)ブロックを初期化した時刻
(7)ブロック領域名

【００２７】これらのうち、まず(1)のフラグについて説明する。フラッシュ管理マネージャ２６（図２参照）は、各物理ブロックヘッダの内容を読み、フラッシュメモリがどのように管理されているかを把握する。フラッシュメモリのメモリセル内のビットデータは、論理値の「１」→「０」への変化の処理時間は比較的短いが、「０」→「１」への変化は、一度ブロックイレース又はイレースをしなければならず、長時間を要する。つまり、１ビットのデータの書換えを行なうために、そのブロック内のデータを一時待避させるとともに、ブロックイレース後に書き戻す必要があり、処理時間としては大きい。

【００２８】そこで、実際には、状態を示すフラグを利用して物理ブロックの状態を管理することになる。すなわち、上述した書換えに伴うオーバーヘッドを軽減するため、物理ブロックヘッダ内にその物理ブロックが有効な状態か，それとも無効な状態か，初期化が完了した状態か，有効な状態から無効な状態への遷移状態か，といった状態をそれぞれの物理ブロック毎に認識し、データの書込み，データの無効化，物理ブロックのイレース，物理ブロックの初期化などの操作を行なう。

【００２９】次に、(2)のマーカーフラグは、該当する物理ブロックがデータファイル領域に割り当てられたか、それとも、実行プログラム領域に割り当てられたかを示すフラグである。これによって、実行プログラム領域に指定されたとき、(3)の連続予約ブロック数に実行プログラムとして後に続くブロックの数を指定する。しかし、データファイル領域に指定されているときは、連続予約ブロック数を０とする。

【００３０】次に、(4)のブロックの消去回数には、各ブロックを消去（ブロックイレース）した回数を記録する。ブロックイレースするときは、記録されている以前の消去回数を一時的に保持し、ブロックイレース終了後、一時的に保持している以前の消去回数に１インクリメントしてデータを格納する。

【００３１】次に、(5)のブロックの消去した順番を示す番号は、同じときにどの順番に物理ブロックのイレースを行なったかを判別するためのものである。ある物理ブロックをブロックイレースし、その後、他の物理ブロックのブロック消去回数が同じものがあれば、その消去回数のブロックのイレースシーケンスナンバーに１インクリメントしたものを該当ブロックヘッダのイレースシーケンスナンバーに格納する。すなわち、消去回数が同じものが複数ブロック存在するとき、それらの中の最大のイレースシーケンスナンバの値に１インクリメントして該当ブロックヘッダ内のイレースシーケンスナンバに格納する。

【００３２】次に、(6)のブロックを初期化した時刻には、該当する物理ブロックをブロックイレースしたときの時刻を記入する。なお、リアルタイムクロックを持たないシステムの場合は不要である。

【００３３】次に、(7)のブロック領域名は、実行プログラムやデータファイルの中でそのブロックに割り当てた固有の名称である。

【００３４】新たにブロックの割り当てを行なう際、全体のブロックが効率よく、つまり特定のブロックに偏らず、消去回数がすべてのブロックで均一化されるように管理するために、上述した管理情報が利用される。まず、フラッシュ管理マネージャ２６は、新たに物理ブロックを割り当てるとき、消去回数の少ないものに対して優先的に割り当てを行なう。もし、消去回数が同じブロックがあるとき、その中でイレースシーケンスナンバのもっとも若い番号のものを割り当てる。最も若いイレースシーケンスナンバのブロックは、消去した時刻が最も古いものである。また、リアルタイムクロックを持つシステムでは、イレースシーケンスナンバの代わりにそのクロックを使用する。すなわち、消去回数が同じブロックがあるとき、その中の物理ブロック内のもっとも古い時刻を持つブロックを割り当てる。このように、イレースシーケンスナンバあるいはリアルタイムクロックのうちのいずれか一方をもてば、少しでも効率よくブロックの割り当てを均一化することができる。

【００３５】実行プログラムの配置は、物理ブロックヘッダの後からその領域内において、実行プログラムデータをＲＯＭの場合と同じように配置することによって行なう。これは、ＯＳ２０に依存する部分が大きいため、従来の方法によって対応することになる。

【００３６】一方、データファイルの配置は、フラッシュ管理マネージャ２６によって行われる。図６［Ａ］に示すように、データファイルは、物理ブロックヘッダ４０Ａの領域を除く記憶領域４０Ｂに格納されることになる。データの格納構造として、データファイル内の論理的なデータレコードをブロッキングし、図６［Ｂ］に示すように、データレコードヘッダ４２Ａ，データレコードフッダ４２Ｃを付随させた形でデータレコード４２Ｂのパケット４２を構成する。ブロッキングの方法としては、固定長レコード，可変長レコード，スパンドレコードがあり、本形態ではいずれの手法であってもよい。このブロッキングは、ＯＳ２０やアプリケーションソフト２２に依存することが多いため、システム上、フラッシュ管理マネージャ２６がブロッキングの方法を決定して制御する。

【００３７】また、データレコードヘッダ４２Ａ，フッダ４２Ｃには、線形リスト構造を持たせるようにする。データレコードの方法によっては、ブロック領域名の最適な利用方法が異なる。システム設計する際、ブロック領域名として、パーティション名，データファイル名，ディレクトリ名などのように、最適なものを割り当てるようにする。最適なブロック領域名は、データレコードの方法に依存することが多い。

【００３８】データファイルの論理的なデータレコード４２Ｂは、上述したようにデータレコードパケット４２として、割り当てられた物理ブロックのデータファイル領域４０Ｂに格納される。図６［Ｂ］にはその様子が示されており、データレコードパケット(1)～(n)が、ｉ－ｊバイトのデータファイル領域４０Ｂ中に格納される。フラッシュ管理マネージャ２６は、データレコードパケット４２の管理と物理ブロック４０Ｂによる管理という２重の管理を行なう。

【００３９】次に、物理ブロックの初期化の手順について説明する。フラッシュ管理マネージャ２６は、フラッシュメモリが初期化されていなければ、まず最初に各物理ブロックを初期化する。初期化する場合は、一度すべてのブロックをイレースし、各物理ブロックのブロックヘッダに初期化完了フラグ，消去回数（１回），イレースシーケンスナンバあるいは、初期化時刻などの情報を格納する。次に、通常は、実行プログラムやデータファイルを格納するため、各物理ブロックの領域を宣言（確保）する。つまり、実行プログラム領域やデータファイル領域を示すマーカーフラグ，連続予約ブロックの領域数の情報を格納する。このときに、ブロックの領域名を割り当るようにしてもよい。また、後でブロックの領域名を割り当てるのであれば、フラッシュ管理マネージャ２６がその対応を行なう。

【００４０】次に、データファイルを格納するときの手順を説明する。物理ブロックヘッダの有効フラグをＯＮにし、その後ブロッキングされたデータレコードをデータ領域内に順次格納して行く。データレコードはデータ領域に追記していくことになるが、データレコードを書き換えたい場合は、フラッシュメモリの特徴からその部分を書き換えることはオーバーヘッドが大きい。

【００４１】そこで、旧論理データレコードから新論理データレコードへのデータ格納は、旧データレコードを無効とし、新たにデータレコードを追記していく方法で行う。無効となったデータレコードは、データレコードヘッダ内に無効フラグを持たせることによって、フラッシュ管理マネージャ２６が無効と判断できる。物理ブロックのデータ領域内のデータレコードパケットのすべてが無効となった場合は、その物理ブロックヘッダ内の無効フラグをＯＮすることによって、フラッシュ管理マネージャ２６はこの物理ブロックをブロックイレースすることができる。ブロックイレースが完了すると、前述のブロックヘッダの初期化を行なう。

【００４２】次に、実行プログラムを格納するときの手順を説明する。最初に実行プログラム領域として複数ブロックを確保するときは、まず、物理ブロックヘッダ内の有効フラグ，実行プログラム領域を示すマーカーフラ

グ，予約領域ブロック数の各情報を格納する。これにより、実行プログラム領域内に実行プログラムをＲＯＭ化したデータを格納する。

【００４３】以上をシステムとして運用し、その途中結果として、データファイル領域の書換回数と実行プログラム領域の書換回数と比較する。そして、大幅な回数の開きがあれば、現在配置されている領域を変更し、デフラグメンテーションを行なうことが望ましい。フラッシュ管理マネージャ２６は、それぞれの領域を管理しているためデフラグメンテーション操作における動的配置に対応でき、システムとして矛盾を起こす心配もない。これらを行なうことで効率的な領域配置を実現できる。また、結果として運用時間を延ばし、フラッシュメモリの書換寿命（回数ではなく時間）を延ばすこともできる。なお、フラッシュメモリによっては、ライト（書込）操作中に他のオペレーションができないチップが存在する。このようなときは、タスク，スレッドのようなプログラミング技法により、排他制御を行なうようにする。

【００４４】図７には、フラッシュメモリに対する物理ブロックヘッダの格納例が示されている。図７［Ａ］の例は、６４Ｋバイトを消去ブロック単位とするフラッシュメモリで、ブロック数がｎ＋１個あるシステムである。各物理ブロックの先頭には、図７［Ｂ］に示すように、各物理ブロックの管理情報であるヘッダ領域として、２５６バイトが割り当てられている。上述したように、データファイルの場合には、該当する物理ブロック内に必ず物理ブロックヘッダを付随させる。しかし、実行プログラムファイルの場合はその限りではない。以下、本形態の実施例について説明する。

【００４５】（1）実施例１

この実施例１は、ブロックシーケンスナンバを用いる例である。本例のファイル管理装置では、図８［Ａ］に示すように、ブロック消去が６４Ｋバイト単位で可能な２Ｍ×８ビットのフラッシュメモリが使用される。図８［Ａ］に示すように、各ブロックに「０」から「１Ｆ」と順に番号を付ける。このようなフラッシュメモリに対して、図５［Ａ］のように実行プログラム領域を連続領域として確保したとすると、図８に［Ｂ］に示すようになる。当然、実行プログラム領域(1)に領域指定されている物理ブロック１～７には物理ブロックヘッダはない。同様に、実行プログラム(2)に領域指定されている物理ブロック１０～Ｅには、物理ブロックヘッダはない。

【００４６】そこで、本実施例では、図９に示すように、１バイト目から１８バイト目までを共通のデータ構造をもつことにする。これにより、この共通部分をフラッシュ管理マネージャ２６によってスキャンすれば、全体の構造を把握することができる。そして、次の１９バイト目から実行プログラム領域あるいはデータファイル領域に割り当てられた内容を追記していけばよい。

【００４７】共通部分の内容を順に説明すると、まず１バイト目には、図１０［Ａ］に示すように、有効／遷移中／無効，初期化完了状態，予約のフラグがある。これらのうち、有効／遷移中／無効のフラグは、このブロック領域が有効状態か，それとも有効から無効になる遷移中（過渡期）の状態か，完全に無効になった状態かを示す。初期化完了状態のフラグは、ブロックのイレース後にブロックヘッダの初期化（フォーマット）が完了したことを指し示す。予約のフラグは、システムを拡張するときに使用するための予備フラグである。実際のビットの割付例を示すと、表１［Ａ］のようになる。

【００４８】

【表１】

[A]

フラグ内容		0	1
有効	ビット７	有効状態	初期化状態
無効	ビット６	無効状態	初期化状態
遷移	ビット５	遷移中状態	初期化状態
初期化	ビット４	初期化完了状態	ブロック消去直後
予約	ビット０から３	－	初期化状態

[B]

		実行プログラム領域	データファイル領域
プログラム／データ	ビット７	0	1

[C]

	ビット０からビット６までの値
ビット７が０ｂのときのみ	連続予約ブロック数（HEX）
データファイル領域　　あるいは 実行プログラム領域が１ブロックのみ	０００００００ｂ

[D]

５バイト目	４バイト目	３バイト目
消去回数（上位）	消去回数（中位）	消去回数（下位）

【００４９】次に、図１０［B］を参照してブロックイレースの手順を説明する。フラッシュ管理マネージャ２６の判定により、ブロックイレース命令が出されたとき（図１０［B］の５０）、次の状態として旧ブロックヘッダの消去回数などを一時保存し（５２）、このブロックのブロックイレースを行なう（５４）。ブロックイレースを行なった直後、イレースされたブロックの記憶エリアのデータ値は、すべてＦＦｈとなっている（５６）。この後すぐに、フラッシュ管理マネージャ２６が初期化（フォーマット）操作（５８）によって、ビット４が０となり、この記憶エリアのデータ値はＦＦｈからＥＦｈに遷移する（６０）。

【００５０】更にその後、このブロックのヘッダに、イレース前に一時保存しておいた消去回数に＋１した値を書き込み、物理ブロックの初期化を行なう。そして、新たにこのブロックへの使用要求によってフラッシュ管理マネージャ２６が割り当てを行ったとすると（６２）、ビット７に０を書き込み、このエリアのデータの値はＥＦｈから６Ｆｈに遷移する（６４）。更に、フラッシュ管理マネージャ２６がこのブロックの無効要求を認めれば（６６）、ビット６に０を書き込み、このエリアのデータの値は６Ｆｈから２Ｆｈに遷移する（６８）。

【００５１】デフラグメンテーションを行なう際、フラッシュ管理マネージャ２６の機能により、物理ブロックにはコピー元，コピー先，ムーブなどのような遷移中の状態が存在する。データファイル領域としての物理ブロック内の有効な状態から遷移中の状態にするには（７０）、ビット５を０にし、このエリアのデータの値は６ＦｈからＦＦｈに遷移させればよい（７２）。この遷移中の状態を存在させることは、遷移に要する時間が少なくないことの他に、電池の消耗による動作の不安定や、カード型フラッシュメモリが外されたときなどにおける安全性を高めるためである。つまり、本発明では、この遷移段階の状態を管理し、故障・事故などによるデータの消失を最小限に抑え、復旧させるように構成されている。

【００５２】次に、図９に示す共通部分の２バイト目について説明する。この２バイト目には、図９に示すように、プログラム／データ，連続予約ブロック領域数のフラグがあり、図１０［C］，表１［C］に示すようになっている。これらのうち、プログラム／データのマーカーフラグは、実行プログラム，データファイルのいずれがそのブロックに割り当てられたかを決定するフラグである。このマーカーフラグが０のとき、実行プログラム領域にこのブロックが指定されたことになる。

【００５３】連続予約ブロック領域数のフラグは、実行プログラム領域に指定した残りのブロック数が１６進数で表記されている。しかし、このマーカーフラグが１で、この物理ブロックはデータファイル領域として割り当てているとき、連続予約ブロック数には０００００００ｂを書く。

【００５４】次に、図９の３～５バイト目は、表１［D］に示すように、そのブロックの消去回数を１６進で表記している。３バイト目がその回数の下位の桁、４バイト目がその回数の中位の桁、５バイト目にはその回数の上位の桁として、それぞれデータを格納する。従っ

て、ＦＦＦＦＦＦｈ回がブロック連続の最大数値となる。なお、この最大値のとき、更なるインクリメントは行われない。

【００５５】次に、図９の６バイト目であるが、これは、各物理ブロックの消去回数が同じときに、フラッシュ管理マネージャ２６が最適な物理ブロックの割り当てを行なうために用意されたもので、上述したイレースシーケンスナンバである。このイレースシーケンスナンバは、物理ブロックの消去回数が他の物理ブロックと同じときに、どの順番で物理ブロックの初期化がなされたかを記録するものである。イレースシーケンスナンバーの割付の流れとして、まず該当する物理ブロックの消去回数と他のブロックで消去回数で同一のものがなければ、該当する物理ブロックのイレースシーケンシナンバは０ｘ００となる。

【００５６】もし、他のブロックで消去回数が同じものがあるときは、その中で最も大きいイレースシーケンスナンバを検出し、該当ブロックのイレースシーケンスナンバには先ほど検出したイレースシーケンスナンバの値に１インクリメントした値を記録する。このようなアルゴリズムによって、イレースシーケンスナンバの割付を行なう。イレースシーケンスナンバは、フラッシュ管理マネージャ２６が物理ブロックの割り当てを行なうときの判断材料となる。

【００５７】次に、図９の７バイト目から１８バイト目までは、該当するブロックの領域名を記録する。実行プログラムファイルの場合、パーティション名やディレクトリ名などに利用すると、システム運用が容易になる。また、データファイルの場合は、データレコード方式によって最適となるようなブロック領域名の扱いが考えられる。例えば、データレコードの方法が固定長，可変長レコード，スパンドレコードのようなファイルを二重管理する場合には、パーティション名，ディレクトリ名などが適している。しかし、固定長や可変長レコードでファイルを一元管理する場合には、データファイル名として扱うほうが望ましい。本実施例では、ブロック領域名は８．３形式で、ＭＳ－ＤＯＳのような表現方法を利用する。

【００５８】以上のような構造の物理ブロックヘッダを用いてフラッシュファイル管理システムを実現する。フラッシュ管理マネージャ２６は、ＯＳ２０やアプリケーションソフト２２などの要求により、実行プログラムやデータファイルなどの読み出し，書き込みを制御する。まず、書き込みに際しては、フラッシュメモリの特徴として書き込み遅延の影響がある。このため、オーバヘッドを少なくして効率よく書き込みを行なうためには、他のデータファイル領域に新たに追記していく方式をとることになる。この方式を実行するためには、書換前の旧データと書換後の新データが論理的には変更されているが、物理的には旧データが無効となり、新データが追記されることになる。

【００５９】つまり、物理的には旧データと新データは共存するものの、旧データに無効フラグを記すことによって論理的に削除することになる。このような動作をフラッシュファイルシステムが繰り返すと、論理的な無効データが数多く存在することになる。場合によっては、新規のデータファイル領域がなくなる危険性すらある。このことを避けるために、フラッシュ管理マネージャ２６は、データの無効化を定期的に検知し、デフラグメンテーションの操作を行なって効率よくデータファイル領域を確保する。

【００６０】データファイル領域としての物理ブロックの割り当ては、フラッシュ管理マネージャ２６が行なう。このとき、ＯＳ２０やアプリケーションソフト２２の要求によってデータファイル領域を与えることになるが、本実施例のアルゴリズムでは、優先的に消去回数が少ない物理ブロックに割り当てられる。また、デフラグメンテーション操作を行なうときなどは、消去回数が少ないブロックについて優先的にデータレコードパケットの無効化を行ない、その物理ブロックのデータファイル領域内をすべて無効化することによって、ブロックの使用回数の均一化が図られる。そしてそれらの中で少しでも物理ブロックの使用回数を効率的に均一化するため、イレースシーケンスナンバの若いものを優先させて、割り当てる物理ブロックやブロックイレースの管理が行われる。

【００６１】フラッシュファイルシステムを運用していくと、結果として実行プログラム領域とデータファイル領域の消去回数が大きく違ってくるようになる。そして、ある程度消去回数に開きが生じると、フラッシュ管理マネージャ２６は、実行プログラム領域で使用している物理ブロックとデータファイル領域で使用している物理ブロックとを入れ替え、これによってフラッシュファイルシステムの運用寿命の延命化が図られる。

【００６２】読み出しの場合は、フラッシュ管理マネージャ２６がフラッシュメモリの読み出しを行なうことを要求し、許可確認後、ブロッキングされたデータレコードパケットを論理的なデータレコードとして扱うことによって可能となる。また、書き込みの場合、フラッシュ管理マネージャ２６にフラッシュメモリへの書き込みを行なうことを要求し、許可確認後、論理的なデータレコードをブロッキングし、データレコードパケット化して指定エリアに格納する。データの書換えは、物理的にデータレコードパケットを無効化し、新たに追記することによって実現する。ただし、物理的にメモリエリアは縮小する。これらのデータのブロッキング方法は、上述した通りであり、背景技術により実現されている。

【００６３】物理ブロックのデータファイル領域内ですべてのデータが無効化された場合、その物理ブロックヘッダの無効フラグをＯＮとすることによって、その物理

ブロック全体が無効化されたことになる。そして、フラッシュ管理マネージャ２６はこのことを検知し、そのブロックをイレースして初期化操作を始める。また、デフラグメンテーション操作により、コピー元のデータファイルの属する物理ブロックヘッダの遷移中フラグをＯＮにする。そして、必要なデータが全部コピーされたなら、その物理ブロックヘッダの無効フラグをＯＮにし、前述のブロックイレース・初期化操作を行なう。遷移中フラグのＯＮから無効フラグのＯＮまでの間に何らかの原因による異常な状態があり、システムが再起動したときなどには、これらの状態フラグを監視し、解析することによって、どの状態から停止あるいは異常となったかを検出することができる。そして、システムを修復・復旧させるための貴重な情報源となる。

【００６４】フラッシュ管理マネージャ２６は、それぞれの物理ブロックの状態を監視，管理していくことになる。ＯＳ２０やアプリケーションソフト２２の要求に応じて該当する物理ブロックを割り当て、有効→（遷移中）→無効→初期化の状態の監視，管理を実行する。そして、フラッシュファイルシステムを運用していくと、各物理ブロックの状態の変化として、初期化→有効，有効→無効，有効→遷移中，移中→無効，無効→初期化，初期化→無効などが挙げられる。

【００６５】［初期化→有効］は、物理ブロックが初期化状態から、その物理ブロックがデータファイル領域等として割り当てられたとき、有効フラグＯＮによって有効状態となる。

【００６６】［有効→無効］は、データファイルとしてその物理ブロックが有効でデータファイル領域内に少なくとも一つは論理的に有効なデータがある状態から、データファイル領域内がすべて論理的に無効なデータとなり、フラッシュ管理マネージャ２６が、その後この物理ブロックの無効フラグをＯＮとし、物理ブロックの無効化を行なう。

【００６７】［有効→遷移中］は、少なくとも一つは存在する論理的に有効なデータレコードパケットを、異なる物理ブロックのデータファイル領域にコピーするとき、コピー元となる物理ブロックに対して遷移中フラグをＯＮにする。この状態のとき、データレコードパケットがコピーされていることになる。

【００６８】［遷移中→無効］は、前述の遷移中の状態が終了し、物理ブロックが無効となったことを示す。つまり、データレコードパケットのコピーが完了し、フラッシュ管理マネージャ２６が無効フラグをＯＮにする。

【００６９】［無効→初期化］は、物理ブロックが無効な状態からその物理ブロックをブロックイレースし、物理ブロックヘッダを初期化が完了したとき、フラッシュ管理マネージャ２６が初期化フラグをＯＮにする。このとき、物理ブロックヘッダの初期化が完了し、データファイル領域として使用可能な状態となる。

【００７０】［初期化→無効］は、物理ブロックが初期化された状態から、その物理ブロックが無効になった状態を示す。これは、状態として存在する可能性はあるが、システムとしては多用すべきではない。

【００７１】フラッシュ管理マネージャ２６は、各物理ブロックの状態を管理・監視し、何らかの原因による異常状態からの脱却後、有効／無効／遷移中／初期化フラグを、異常状態以前のデータの復元や復旧をするための情報源とする。つまり、これらの変化状態を考慮することで、システムの停止や異常が生じたシステムの修復やデータレコードパケットの復元を行なうとき、どの状態から異常になったかを検出することができる。

【００７２】(2) 実施例２

この実施例２は、実施例１のイレースシーケンスナンバの代わりに初期化した時刻を記録する方法である。すなわち、図１１にブロックヘッダの構成を示すように、１９バイト目から２１バイト目までに初期化が行われた年／月／日／時／分／秒を記録する。実施例１のイレースシーケンスナンバは、同一消去回数のときの初期化された順番を指していたが、実施例２では、ブロックの使用回数の均一化を図るアルゴリズムにおいて、すべての物理ブロックの使用回数を少しでも効率的に向上させるために、消去回数が同じときは最も古い時刻のものを優先させることによって、割り当てる物理ブロックやブロックイレースの管理が行なわれる。

【００７３】(3) 実施例３

この実施例３は、実施例１と実施例２を組み合わせ、イレースシーケンスナンバと初期化時刻の双方を記録する方法である。図１２に示すように、６バイト目にイレースシーケンスナンバを記録し、１９バイト目から２１バイト目までに初期化の年／月／日／時／分／秒を記録する。実施例１，実施例２とほとんど同じであるが、ブロックの使用回数の均一化を図るアルゴリズムにおいて、すべての物理ブロックの使用回数を少しでも効率的に向上させるために、消去回数が同じときはイレースシーケンスナンバの若いものから優先させるようにする。また、最も古い初期化時刻の物理ブロックが存在するときは、消去回数を他と見比べて効率がよいと判断できるとき、その物理ブロックを優先的に割り当てるアルゴリズムによって、割り当てる物理ブロックやブロックイレースの管理が行なわれる。

【００７４】この発明には数多くの実施形態があり、以上の開示に基づいて多様に改変することが可能である。例えば、次のようなものも含まれる。

(1)前記形態はフラッシュメモリに本発明を適用したものであるが、他の類似するメモリがフラッシュメモリと同じ書込み，読出し機能を備えており、かつ、書込前ブロック消去特性を有するメモリであれば、同様に適用可能である。

(2)フラッシュメモリのイレース単位である物理ブロッ

クは、バイト単位やワード単位の他、どのようなデータ単位であっても、同様に適用可能である。

【００７５】

【発明の効果】以上説明したように本発明によれば、次のような効果がある。

(1)フラッショ型メモリを使用した拡張記憶装置上で、フラッシュ型メモリの書換回数を少なくするような制御構造を持ち、かつ、主記憶装置に実行プログラムをロードすることなく、フラッシュ型メモリ上でプログラムの実行が可能となる。また、書換回数を少なくする制御構造をもつ補助記憶装置としても、無駄の少ないデータ構造で記憶を行うことが可能となる。

【００７６】(2)各ブロックの書換回数等の情報テーブルのためのエリアをブロック内に設けることとしたので、別にメモリを設ける必要がなく、コストの削減が可能となる。特に、拡張記憶装置と補助記憶装置を１つのフラッシュ型メモリ上で混在させることができ、システムとして省スペース，コスト削減を図ることができる。

【００７７】(3)物理ブロックの有効／遷移中／無効／初期化という状態フラグをもつこととしたので、コンピュータシステムの異常による停止からの復旧やデータの復元に有効である。

【図面の簡単な説明】

【図１】この発明の一形態を利用したシステムの基本構成を示す図である。

【図２】一形態におけるソフトウエア構造を示す図である。

【図３】一形態におけるフラッシュメモリ内での物理ブロックの割付を行なったメモリマップを示す図である。

【図４】１つの実行プログラム領域と複数のデータファイル領域を物理ブロックに割り当てた例を示す図である。

【図５】複数の実行プログラム領域と複数のデータファイル領域を物理ブロックに割り当てた例を示す図である。

【図６】データファイル領域のデータレコードパケットの例と格納方法の例を示す図である。

【図７】ブロックイレースが６４Ｋバイト単位のフラッシュメモリを物理ブロック単位に分割して２５６バイトのブロックヘッダを確保した例を示す図である。

【図８】実施例１で使用しているフラッシュメモリを実行プログラム領域とデータファイル領域に分割したメモリマップの例を示す図である。

【図９】実施例１で使用している物理ブロックヘッダのメモリマップの例を示す図である。

【図１０】実施例１における物理ブロックヘッダの一部とシーケンスを表す図である。

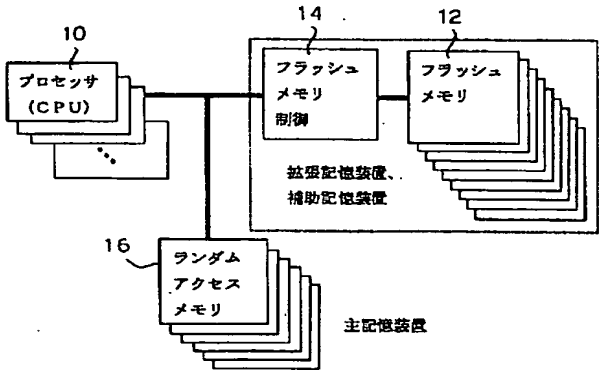
【図１１】実施例２で使用している物理ブロックヘッダのメモリマップの例を示す図である。

【図１２】実施例２で使用している物理ブロックヘッダのメモリマップの例を示す図である。

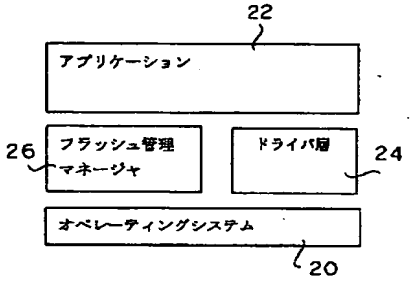
【符号の説明】

１０…プロセッサ
１２…フラッシュメモリ
１４…フラッシュメモリ制御装置
１６…ＲＡＭ
２０…オペレーティングシステム
２２…アプリケーションソフト
２４…ドライバ層
２６…フラッシュ管理マネージャ
３０Ａ，３１Ａ，３８Ａ，３９Ａ，３（ｎ－１）Ａ，３ｎＡ，４０Ａ…ヘッダ領域
３０Ｂ，３１Ｂ，３８Ｂ，３９Ｂ，３（ｎ－１）Ｂ，３ｎＢ，４０Ｂ…記憶領域
３２…実行プログラム領域
４２…データレコードパケット
４２Ａ…データレコードヘッダ
４２Ｂ…データレコード
４２Ｃ…データレコードフッダ

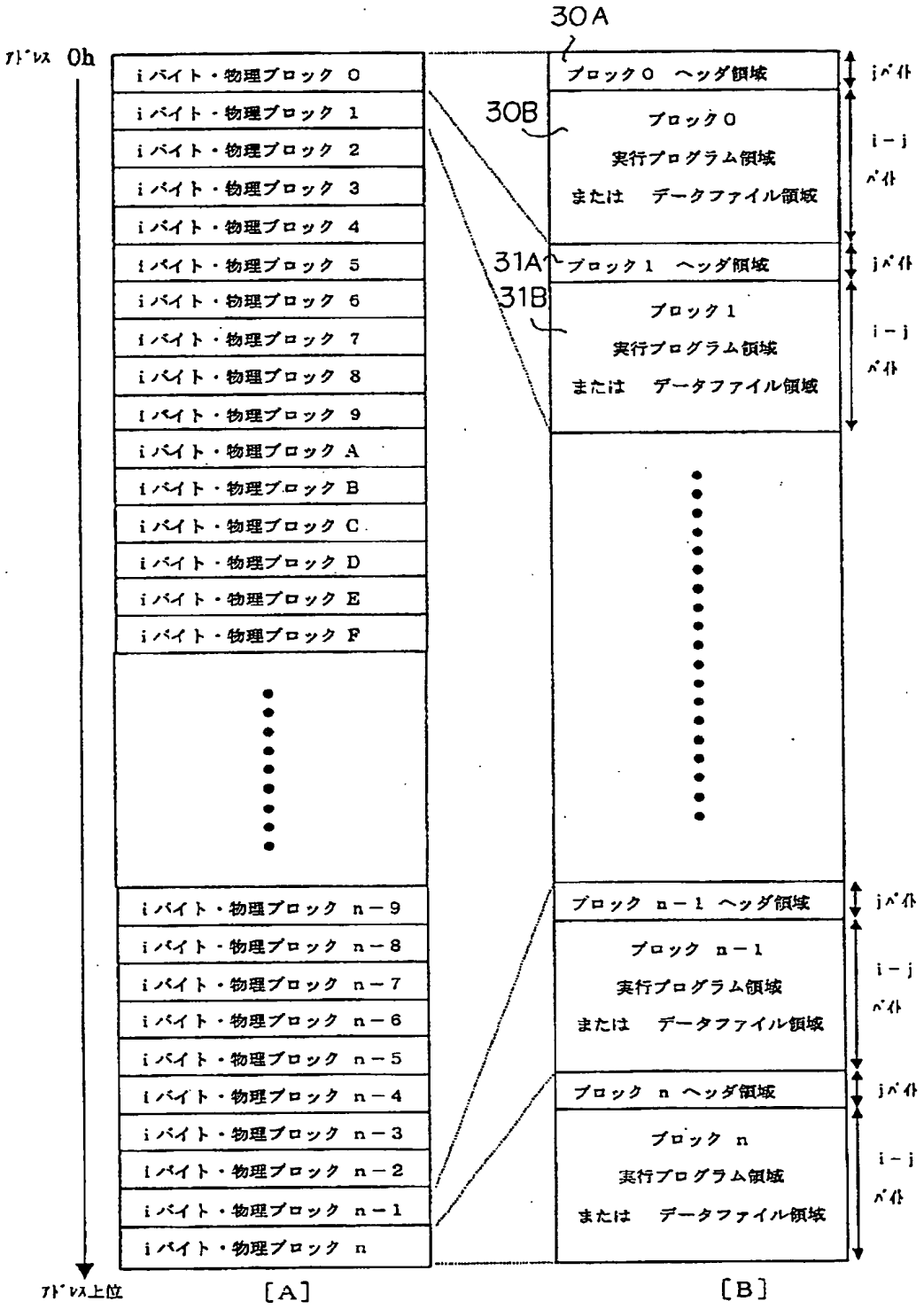
【図１】



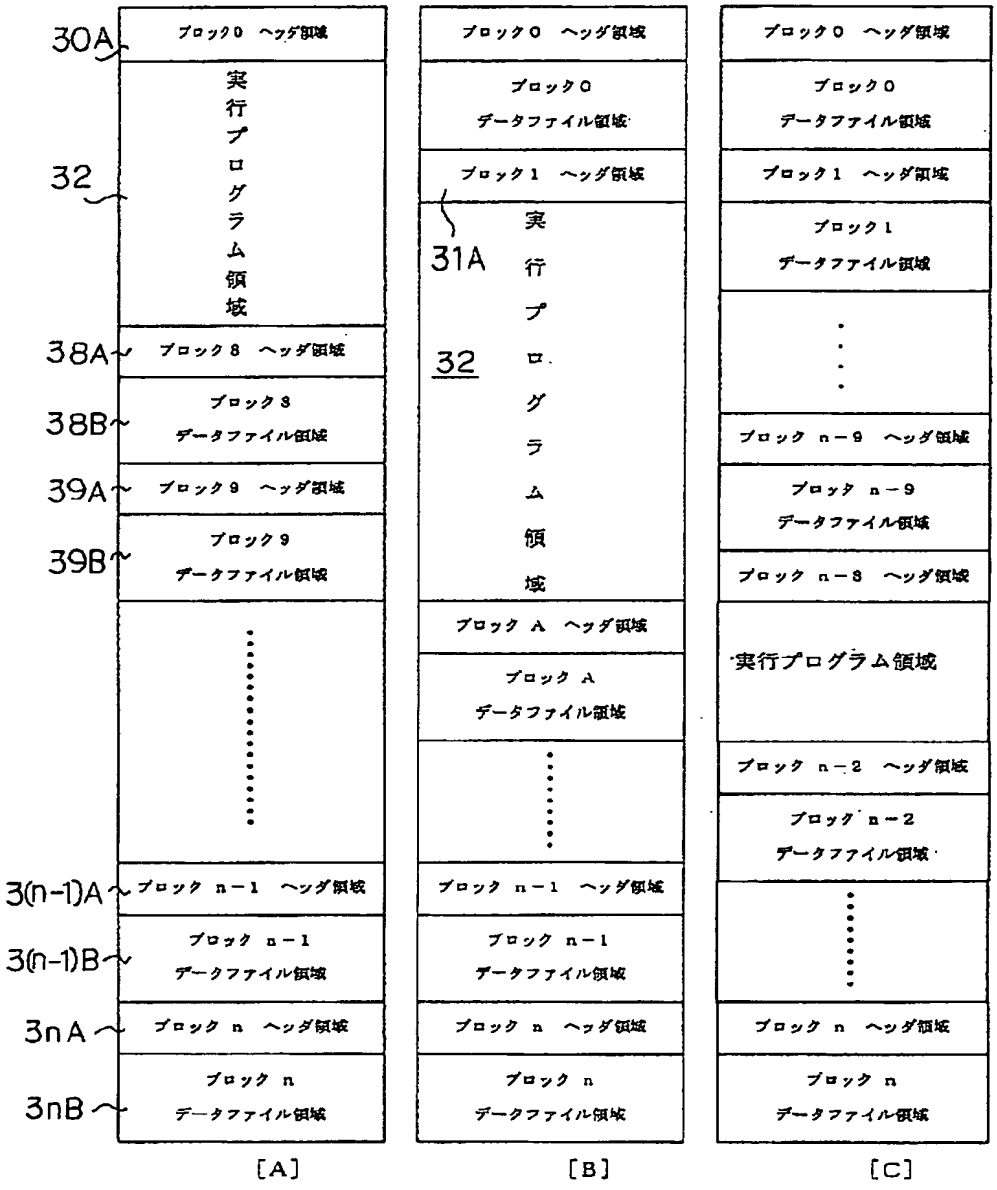
【図２】



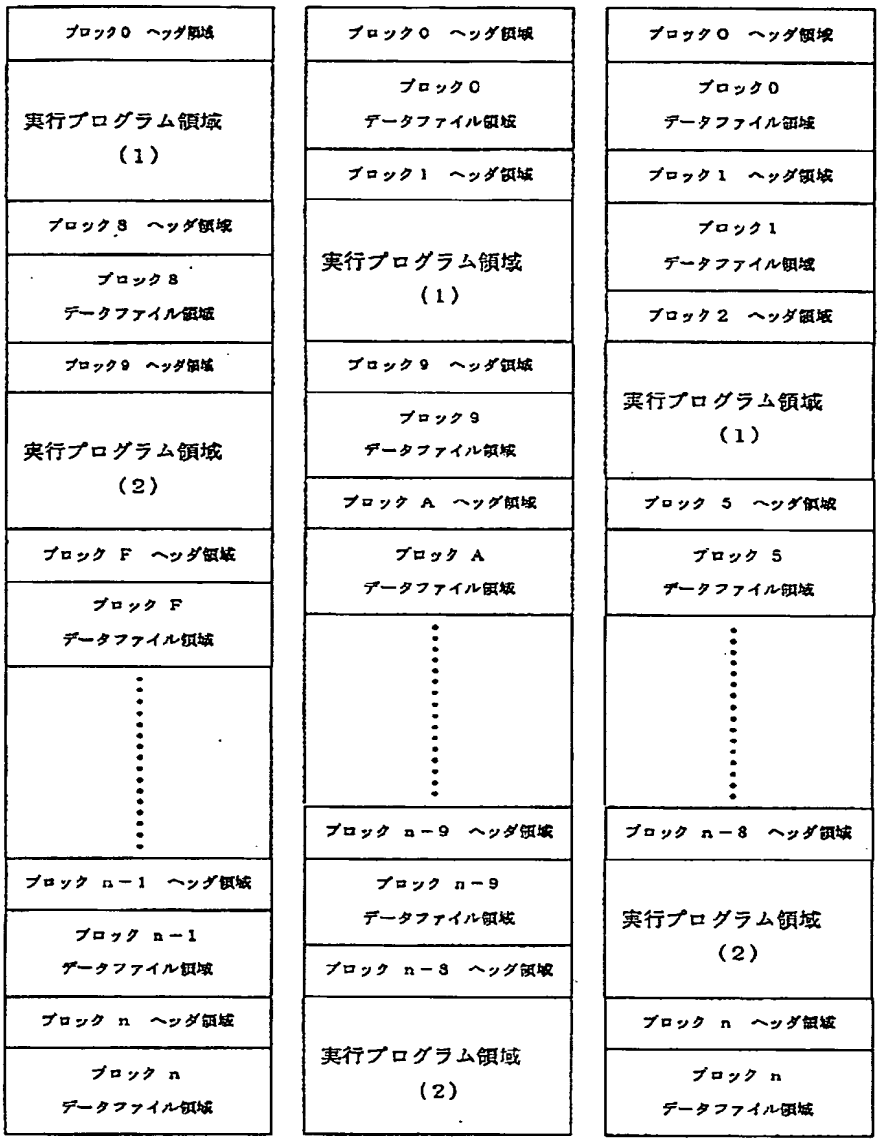
【図３】


30A
ｱﾄﾞﾚｽ 0h
iバイト・物理ブロック 0
iバイト・物理ブロック 1
iバイト・物理ブロック 2
iバイト・物理ブロック 3
iバイト・物理ブロック 4
iバイト・物理ブロック 5
iバイト・物理ブロック 6
iバイト・物理ブロック 7
iバイト・物理ブロック 8
iバイト・物理ブロック 9
iバイト・物理ブロック A
iバイト・物理ブロック B
iバイト・物理ブロック C
iバイト・物理ブロック D
iバイト・物理ブロック E
iバイト・物理ブロック F
iバイト・物理ブロック n－9
iバイト・物理ブロック n－8
iバイト・物理ブロック n－7
iバイト・物理ブロック n－6
iバイト・物理ブロック n－5
iバイト・物理ブロック n－4
iバイト・物理ブロック n－3
iバイト・物理ブロック n－2
iバイト・物理ブロック n－1
iバイト・物理ブロック n
ｱﾄﾞﾚｽ上位
[A]
30B
31A
31B
ブロック０ ヘッダ領域
ブロック０
実行プログラム領域
または データファイル領域
ブロック１ ヘッダ領域
ブロック１
実行プログラム領域
または データファイル領域
ブロック n－1 ヘッダ領域
ブロック n－1
実行プログラム領域
または データファイル領域
ブロック n ヘッダ領域
ブロック n
実行プログラム領域
または データファイル領域
jﾊﾞｲﾄ
i－j
ﾊﾞｲﾄ
[B]

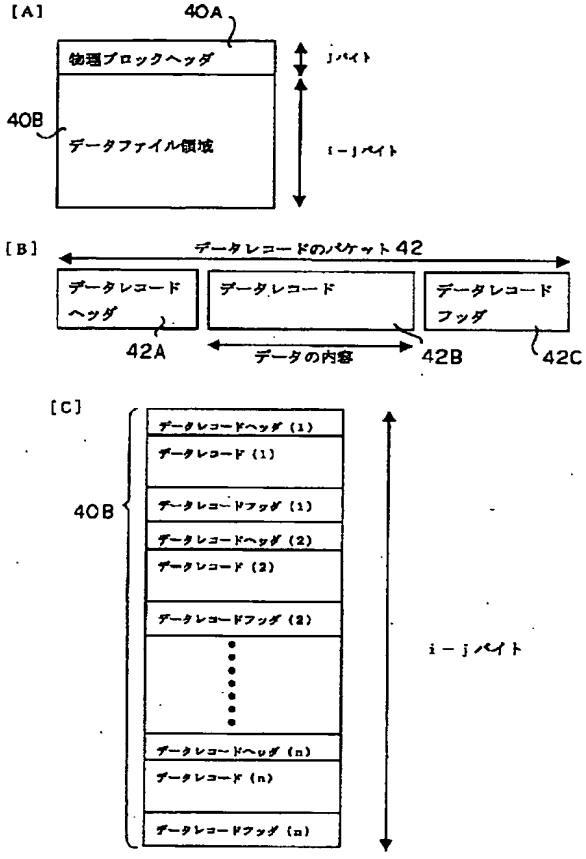
【図４】


30A
32
38A
38B
39A
39B
3(n-1)A
3(n-1)B
3nA
3nB
ブロック０ ヘッダ領域
実行プログラム領域
ブロック８ ヘッダ領域
ブロック８ データファイル領域
ブロック９ ヘッダ領域
ブロック９ データファイル領域
ブロック ｎ－１ ヘッダ領域
ブロック ｎ－１ データファイル領域
ブロック ｎ ヘッダ領域
ブロック ｎ データファイル領域
ブロック０ データファイル領域
ブロック１ ヘッダ領域
31A
ブロック Ａ ヘッダ領域
ブロック Ａ データファイル領域
ブロック １ データファイル領域
ブロック ｎ－９ ヘッダ領域
ブロック ｎ－９ データファイル領域
ブロック ｎ－８ ヘッダ領域
実行プログラム領域
ブロック ｎ－２ ヘッダ領域
ブロック ｎ－２ データファイル領域
[A]
[B]
[C]

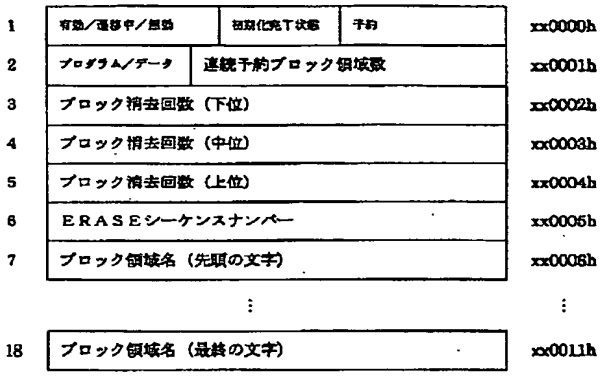
【図５】


ブロック０ ヘッダ領域
実行プログラム領域
（１）
ブロック８ ヘッダ領域
ブロック８
データファイル領域
ブロック９ ヘッダ領域
実行プログラム領域
（２）
ブロック Ｆ ヘッダ領域
ブロック Ｆ
データファイル領域
ブロック ｎ－１ ヘッダ領域
ブロック ｎ－１
データファイル領域
ブロック ｎ ヘッダ領域
ブロック ｎ
データファイル領域
［A］
ブロック０ ヘッダ領域
ブロック０
データファイル領域
ブロック１ ヘッダ領域
実行プログラム領域
（１）
ブロック９ ヘッダ領域
ブロック９
データファイル領域
ブロック Ａ ヘッダ領域
ブロック Ａ
データファイル領域
ブロック ｎ－９ ヘッダ領域
ブロック ｎ－９
データファイル領域
ブロック ｎ－８ ヘッダ領域
実行プログラム領域
（２）
［B］
ブロック０ ヘッダ領域
ブロック０
データファイル領域
ブロック１ ヘッダ領域
ブロック１
データファイル領域
ブロック２ ヘッダ領域
実行プログラム領域
（１）
ブロック ５ ヘッダ領域
ブロック ５
データファイル領域
ブロック ｎ－８ ヘッダ領域
実行プログラム領域
（２）
ブロック ｎ ヘッダ領域
ブロック ｎ
データファイル領域
［C］

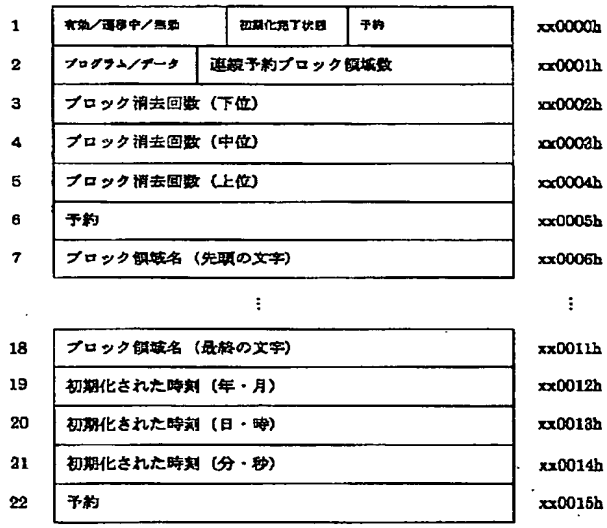
【図６】


[A]
40A
物理ブロックヘッダ
jバイト
40B
データファイル領域
i－jバイト
[B]
データレコードのパケット42
データレコードヘッダ
データレコード
データレコードフッダ
42A
データの内容
42B
42C
[C]
40B
データレコードヘッダ (1)
データレコード (1)
データレコードフッダ (1)
データレコードヘッダ (2)
データレコード (2)
データレコードフッダ (2)
データレコードヘッダ (n)
データレコード (n)
データレコードフッダ (n)
i－jバイト

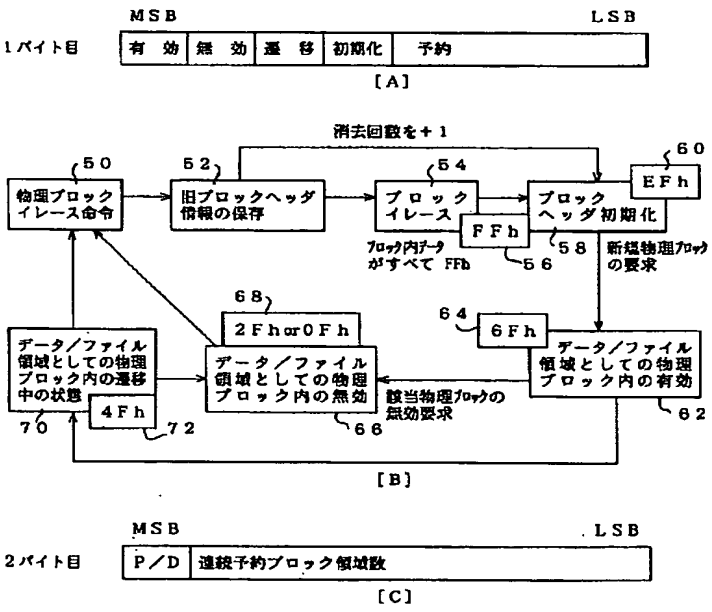
【図９】


1 有効/運移中/無効 初期化完了状態 予約 xx0000h
2 プログラム/データ 連続予約ブロック領域数 xx0001h
3 ブロック消去回数（下位） xx0002h
4 ブロック消去回数（中位） xx0003h
5 ブロック消去回数（上位） xx0004h
6 ERASEシーケンスナンバー xx0005h
7 ブロック領域名（先頭の文字） xx0006h
18 ブロック領域名（最終の文字） xx0011h

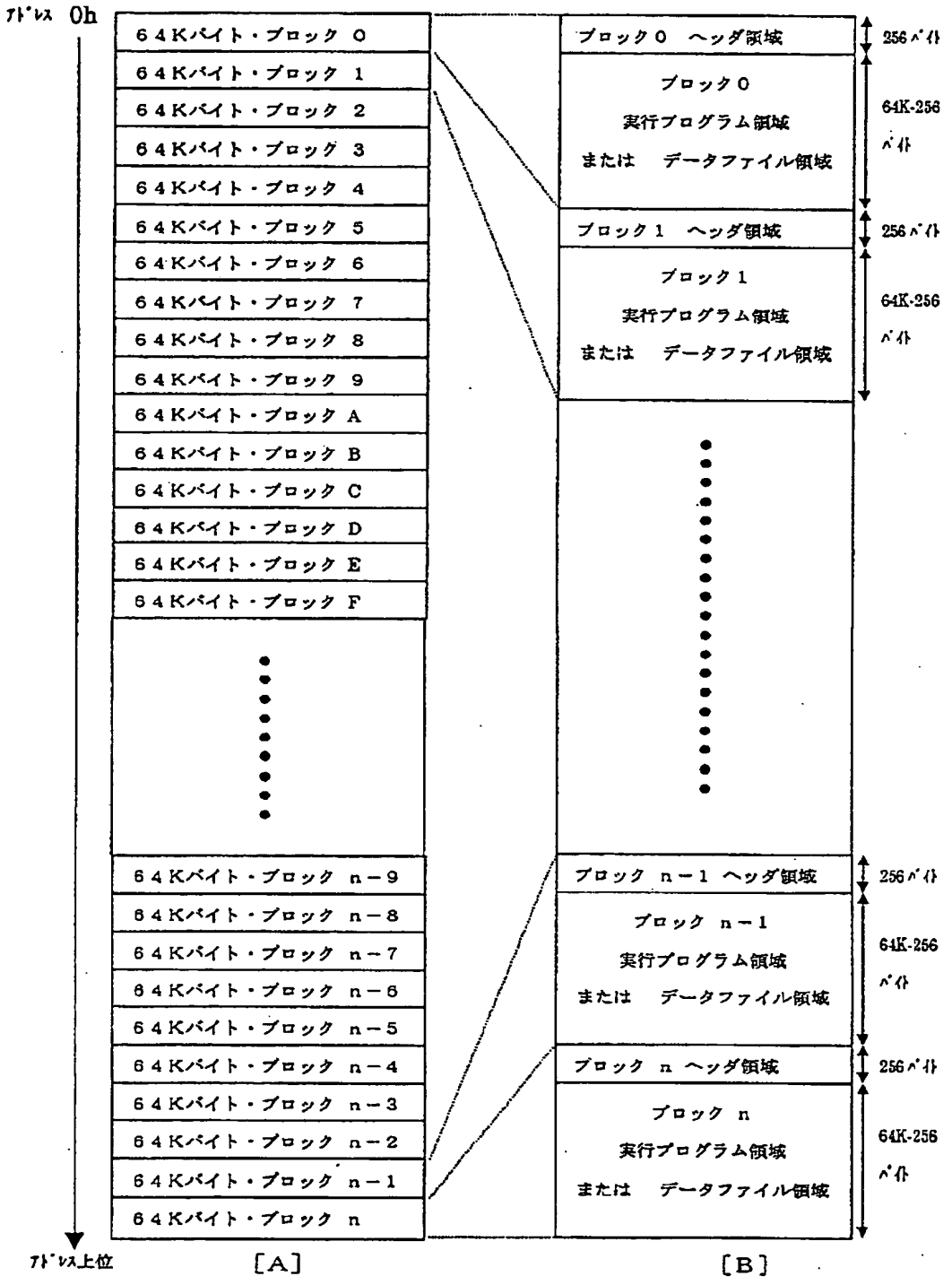
【図１１】


1 有効/運移中/無効 初期化完了状態 予約 xx0000h
2 プログラム/データ 連続予約ブロック領域数 xx0001h
3 ブロック消去回数（下位） xx0002h
4 ブロック消去回数（中位） xx0003h
5 ブロック消去回数（上位） xx0004h
6 予約 xx0005h
7 ブロック領域名（先頭の文字） xx0006h
18 ブロック領域名（最終の文字） xx0011h
19 初期化された時刻（年・月） xx0012h
20 初期化された時刻（日・時） xx0013h
21 初期化された時刻（分・秒） xx0014h
22 予約 xx0015h

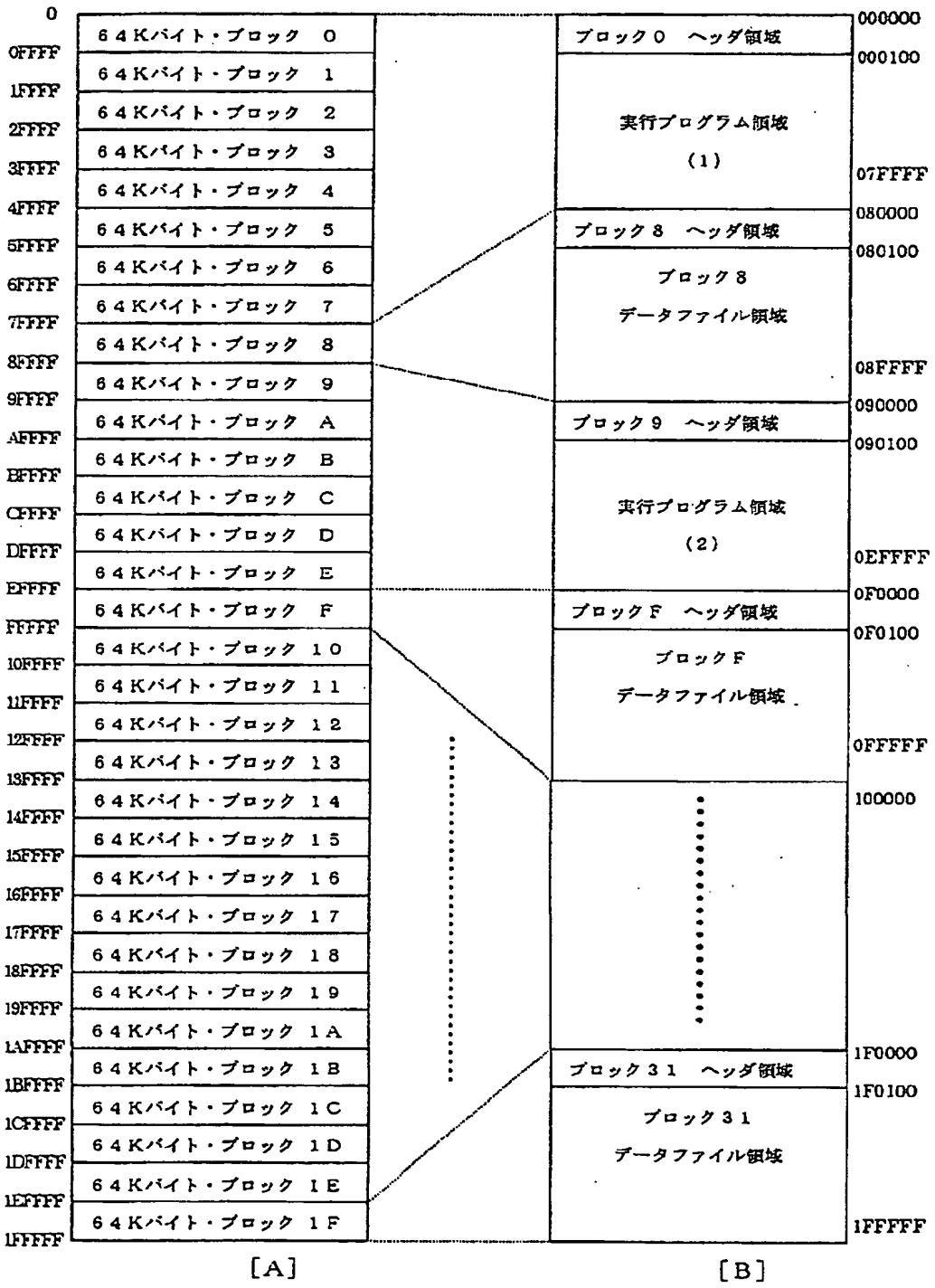
【図１０】


MSB
LSB
1バイト目
有効
無効
遷移
初期化
予約
[A]
消去回数を＋１
50
物理ブロック
イレース命令
52
旧ブロックヘッダ
情報の保存
54
ブロック
イレース
FFh
56
ブロック内データ
がすべて FFh
58
ブロック
ヘッダ初期化
60
EFh
新規物理ブロック
の要求
64
6Fh
データ/ファイル
領域としての物理
ブロック内の有効
62
68
2FhorOFh
データ/ファイル
領域としての物理
ブロック内の無効
66
該当物理ブロックの
無効要求
データ/ファイル
領域としての物理
ブロック内の遷移
中の状態
70
4Fh
72
[B]
MSB
LSB
2バイト目
P/D
連続予約ブロック領域数
[C]

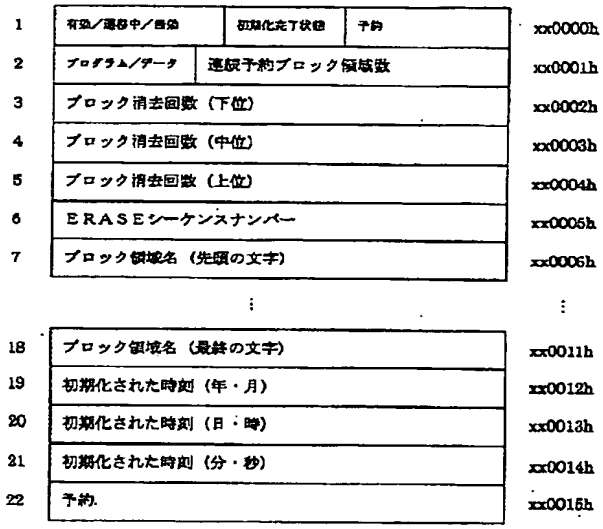
【図７】


ｱﾄﾞﾚｽ 0h
６４Ｋバイト・ブロック ０
６４Ｋバイト・ブロック １
６４Ｋバイト・ブロック ２
６４Ｋバイト・ブロック ３
６４Ｋバイト・ブロック ４
６４Ｋバイト・ブロック ５
６４Ｋバイト・ブロック ６
６４Ｋバイト・ブロック ７
６４Ｋバイト・ブロック ８
６４Ｋバイト・ブロック ９
６４Ｋバイト・ブロック Ａ
６４Ｋバイト・ブロック Ｂ
６４Ｋバイト・ブロック Ｃ
６４Ｋバイト・ブロック Ｄ
６４Ｋバイト・ブロック Ｅ
６４Ｋバイト・ブロック Ｆ
６４Ｋバイト・ブロック ｎ－９
６４Ｋバイト・ブロック ｎ－８
６４Ｋバイト・ブロック ｎ－７
６４Ｋバイト・ブロック ｎ－６
６４Ｋバイト・ブロック ｎ－５
６４Ｋバイト・ブロック ｎ－４
６４Ｋバイト・ブロック ｎ－３
６４Ｋバイト・ブロック ｎ－２
６４Ｋバイト・ブロック ｎ－１
６４Ｋバイト・ブロック ｎ
ｱﾄﾞﾚｽ上位
[A]
ブロック０　ヘッダ領域
256ﾊﾞｲﾄ
ブロック０
実行プログラム領域
または　データファイル領域
64K-256
ﾊﾞｲﾄ
ブロック１　ヘッダ領域
256ﾊﾞｲﾄ
ブロック１
実行プログラム領域
または　データファイル領域
64K-256
ﾊﾞｲﾄ
ブロック　ｎ－１　ヘッダ領域
256ﾊﾞｲﾄ
ブロック　ｎ－１
実行プログラム領域
または　データファイル領域
64K-256
ﾊﾞｲﾄ
ブロック　ｎ　ヘッダ領域
256ﾊﾞｲﾄ
ブロック　ｎ
実行プログラム領域
または　データファイル領域
64K-256
ﾊﾞｲﾄ
[B]

【図８】


0
0FFFF
1FFFF
2FFFF
3FFFF
4FFFF
5FFFF
6FFFF
7FFFF
8FFFF
9FFFF
AFFFF
BFFFF
CFFFF
DFFFF
EFFFF
FFFFF
10FFFF
11FFFF
12FFFF
13FFFF
14FFFF
15FFFF
16FFFF
17FFFF
18FFFF
19FFFF
1AFFFF
1BFFFF
1CFFFF
1DFFFF
1EFFFF
1FFFFF
６４Ｋバイト・ブロック ０
６４Ｋバイト・ブロック １
６４Ｋバイト・ブロック ２
６４Ｋバイト・ブロック ３
６４Ｋバイト・ブロック ４
６４Ｋバイト・ブロック ５
６４Ｋバイト・ブロック ６
６４Ｋバイト・ブロック ７
６４Ｋバイト・ブロック ８
６４Ｋバイト・ブロック ９
６４Ｋバイト・ブロック Ａ
６４Ｋバイト・ブロック Ｂ
６４Ｋバイト・ブロック Ｃ
６４Ｋバイト・ブロック Ｄ
６４Ｋバイト・ブロック Ｅ
６４Ｋバイト・ブロック Ｆ
６４Ｋバイト・ブロック １０
６４Ｋバイト・ブロック １１
６４Ｋバイト・ブロック １２
６４Ｋバイト・ブロック １３
６４Ｋバイト・ブロック １４
６４Ｋバイト・ブロック １５
６４Ｋバイト・ブロック １６
６４Ｋバイト・ブロック １７
６４Ｋバイト・ブロック １８
６４Ｋバイト・ブロック １９
６４Ｋバイト・ブロック １Ａ
６４Ｋバイト・ブロック １Ｂ
６４Ｋバイト・ブロック １Ｃ
６４Ｋバイト・ブロック １Ｄ
６４Ｋバイト・ブロック １Ｅ
６４Ｋバイト・ブロック １Ｆ
[A]
ブロック０ ヘッダ領域
実行プログラム領域
(1)
ブロック８ ヘッダ領域
ブロック８
データファイル領域
ブロック９ ヘッダ領域
実行プログラム領域
(2)
ブロックＦ ヘッダ領域
ブロックＦ
データファイル領域
ブロック３１ ヘッダ領域
ブロック３１
データファイル領域
000000
000100
07FFFF
080000
080100
08FFFF
090000
090100
0EFFFF
0F0000
0F0100
0FFFFF
100000
1F0000
1F0100
1FFFFF
[B]

【図１２】


1
有効／遷移中／無効
初期化完了状態
予約
xx0000h
2
プログラム／データ
連続予約ブロック領域数
xx0001h
3
ブロック消去回数（下位）
xx0002h
4
ブロック消去回数（中位）
xx0003h
5
ブロック消去回数（上位）
xx0004h
6
ＥＲＡＳＥシーケンスナンバー
xx0005h
7
ブロック領域名（先頭の文字）
xx0006h
18
ブロック領域名（最終の文字）
xx0011h
19
初期化された時刻（年・月）
xx0012h
20
初期化された時刻（日・時）
xx0013h
21
初期化された時刻（分・秒）
xx0014h
22
予約
xx0015h